Федеральное государственное бюджетное образовательное учреждение высшего

образования

«Чувашский государственный университет им. И.Н. Ульянова»

Кафедра вычислительной техники

Лабораторная работа № 2

Деревья. Бинарные деревья.

Вариант 9

Выполнил: Иванов В.С.

студент группы ИВТ-41-22

Проверил: кандидат тех. Наук

Павлов Леонид Александрович

Чебоксары, 2023

**Работа №2: Деревья, бинарные деревья.**

***Цель работы*** – Изучить способы узлового представления деревьев и методы их прохождения, получить практические навыки программирования задач с использованием деревьев.

***Выполнение работы:***

1. *Реализовать узловое представление бинарного дерева.*

struct tree:

Data

Tree(left)

Tree(right)

1. *Разработать алгоритмы, реализующие прямое, обратное, симметричное и горизонтальное прохождения бинарного дерева. Для первых трех прохождений использовать как рекурсивные, так и нерекурсивные варианты. Для реализации горизонтального прохождения требуется очередь.*

***Нерекурсивные алгоритмы***

*Прямой обход дерева:*

Procedure preorder(node)

nodestack = empty stack

while (not nodestack.isEmpty() or node != null) do

if (node != null) then

visit(node)

nodestack.push(node)

node = node.left

else

node = nodestack.pop()

node = node.right

*Симметричный обход дерева:*

Procedure inorder(node)

Nodestack = empty stack

While (not nodestack.isEmpty() or node != null) do

if (node != null) then

nodestack.push(node)

node = node.left

else

node = nodestack.pop()

visit(node)

node = node.right

*Обратный обход дерева:*

Procedure postorder(node)

Nodestack = empty stack

Lastnodevisited = null

while (not nodestack.isEmpty() or node != null) do

if (node != null) then

nodestack.push(node)

node = node->left

else

peeknode = nodestack.peek()

if (peeknode->right != null and lastnodevisited != peeknode->right)

then

node = peeknode->right

else:

nodestack.pop()

visit(peeknode)

lastnodevisited = peeknode

*Горизонтальный обход дерева:*

Procedure horizontal(node)

Nodequeue = empty queue

Nodequeue.pushback(node)

While (not nodequeue.isEmpty()) do

Node\* temp = nodequeue.popFront(nodequeue)

Visit(temp)

If (temp.left != null) then

Nodequeue.pushBack(temp.left)

If (temp.right != null) then

Nodequeue.pushBack(temp.right)

***Рекурсивные алгоритмы:***

*Прямой обход дерева:*

Procedure preorder(node)

If (node) then

Print(node.data)

Preorder(node.left)

Preorder(node.right)

*Симметричный обход дерева:*

Procedure inorder(node)

If (node) then

Inorder(node.left)

Print(node.data)

Inorder(node.right)

*Обратный обход дерева:*

Procedure postorder(node)

If (node)

Postorder(node.left)

Postorder(node.right)

Print(node.data)

1. *Разработать алгоритмы и программы решения задач в соответствии с заданными вариантами.*

*Задание:*Вычисление среднего арифметического всех элементов непустого дерева.

*Алгоритм:*

Procedure preorder(node)

nodestack = empty stack

sum = 0

while (not nodestack.isEmpty() or node != null) do

if (node != null) then

visit(node)

sum = sum + node.data

nodestack.push(node)

node = node.left

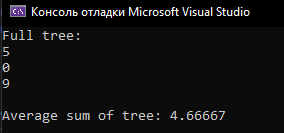
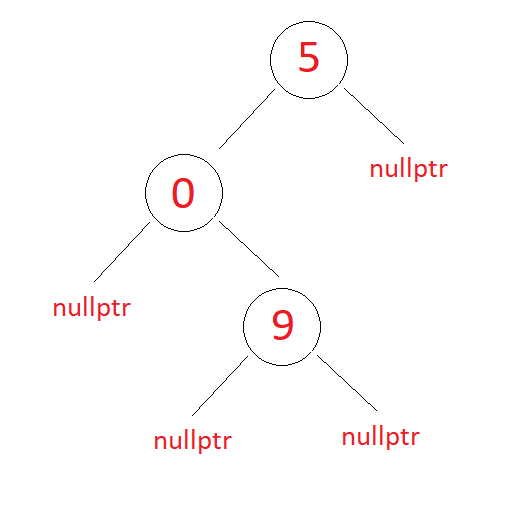
else

node = nodestack.pop()

node = node.right

print(sum)

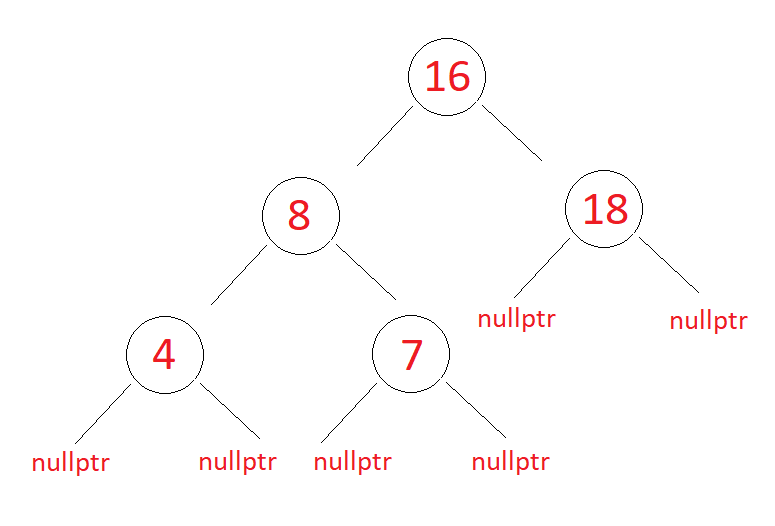
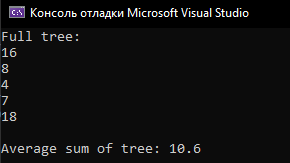
*Пример 1:*



*Анализ вывода программы:*

Среднее арифметическое элементов дерева: (5+0+9)/3= 4,(6)

*Пример 2*:



*Анализ вывода программы:*

Среднее арифметическое элементов дерева: (16+8+4+7+18)/5 = 10,6

Задание: Вывод элементов из всех листьев дерева.

Код программы:

#include<iostream>

#include<vector>

#include<stack>

using namespace std;

struct node

{

int key\_value;

node\* left;

node\* right;

node(int x) : key\_value(x), left(NULL), right(NULL) {}

};

class btree

{

public:

btree();

void insert(int key);

node\* root;

};

vector<int> LeafElement(node\* root) {

vector<int> result;

stack<node\*> nodes;

node\* current = root;

while (current != NULL || !nodes.empty()) {

while (current != NULL) {

nodes.push(current);

current = current->left;

}

current = nodes.top();

nodes.pop();

if (current->left == NULL && current->right == NULL) {

result.push\_back(current->key\_value);

}

current = current->right;

}

return result;

}

int main() {

int a, b, c, d, e, f, g;

cout << "Enter 7 numbers: " << endl;

cin >> a >> b >> c >> d >> e >> f >> g;

node\* root = new node(a);

root->left = new node(b);

root->right = new node(c);

root->left->left = new node(d);

root->left->right = new node(e);

root->right->left = new node(f);

root->right->right = new node(g);

vector<int> leafElements = LeafElement(root);

cout << "Leafes of tree: " << endl;

for (int element : leafElements) {

cout << element << " ";

}

cout << endl;

}

***Вывод -*** Изучил способы узлового представления деревьев и методы их прохождения, получил практические навыки программирования задач с использованием деревьев.